**Business Case: Target SQL**

This business case has data of 100k orders from 2016 to 2018 made at Target, Brazil. It is Americas leading retailer business chain.

Data is available in 8 tables, which gives information about orders from different dimensions like status of order, payment details, location and time of the order , customer who made the purchase, items in the order, product details, seller information of the products, order reviews etc.

**Analysis**

**1. Initial exploration of dataset**

1.1 Show all the tables and all the columns present in each table along with its data type.

Query:

SELECT

  table\_schema,

  table\_name,

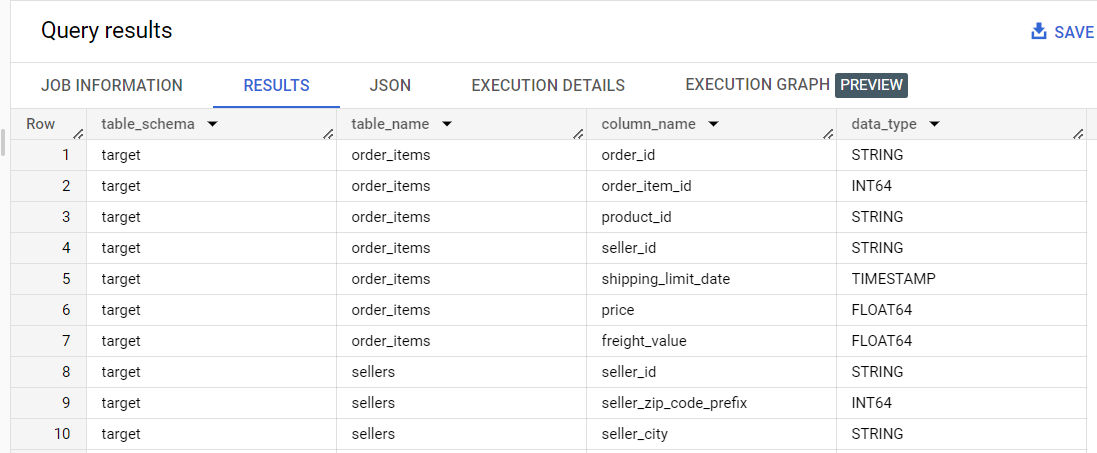
  column\_name,

  data\_type

FROM

  `target`.INFORMATION\_SCHEMA.COLUMNS;

Result:



* 1. For which time period the data is given.

Query:

SELECT

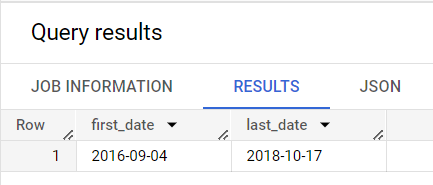
  MIN(DATE(order\_purchase\_timestamp)) AS first\_date,

  MAX(DATE(order\_purchase\_timestamp)) AS last\_date

FROM

  `target.orders`;

Result:



* 1. From which Cities and States , orders were placed during the given period.

Query:

SELECT

  DISTINCT c.customer\_state , c.customer\_city

FROM

  `target.customers` c

RIGHT JOIN

  `target.orders` o

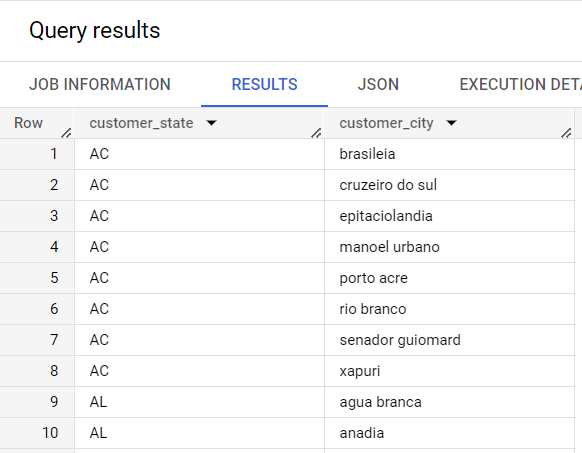
USING

  (customer\_id)

ORDER BY

  c.customer\_state , c.customer\_city;

Result:



* 1. What is distribution of total orders as per their status?

Query:

SELECT

  order\_status,

  COUNT(\*) AS orders\_count

FROM

  `target.orders`

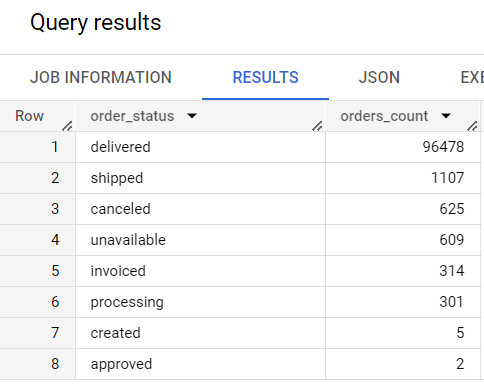
GROUP BY

  order\_status

ORDER BY

  orders\_count DESC;

Result:



* 1. How is the order value of each order is calculated in payment table, Is it addition of price and freight for each item in the order.

*(Used order items and payment table)*

SELECT

  order\_id,   total\_price, total\_freight,

  total\_price + total\_freight AS total\_order\_value

FROM ( SELECT

    order\_id,

    SUM(price) AS total\_price,

    SUM(freight\_value) AS total\_freight,

    COUNT(\*) AS total\_items

  FROM

    `target.order\_items`

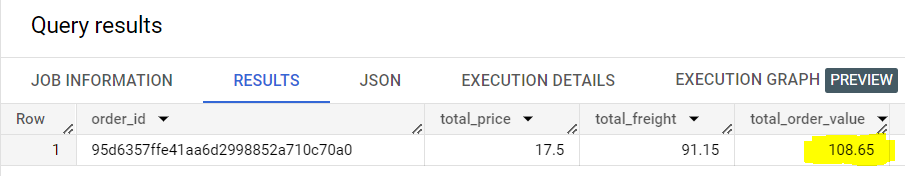
  WHERE

    order\_id = "95d6357ffe41aa6d2998852a710c70a0"

  GROUP BY

    order\_id) t1;

Result:



  SELECT

  order\_id, payment\_value

FROM

  `target.payments`

WHERE

  order\_id = "95d6357ffe41aa6d2998852a710c70a0";



**2. In-depth exploration of dataset**

2.1 Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?

Query:

SELECT

  time\_period,

  order\_count,

  ROUND((((order\_count - LAG(order\_count) OVER(ORDER BY t1.YEAR, t1.month)) / LAG(order\_count) OVER(ORDER BY t1.YEAR, t1.month))\* 100), 2) AS growth\_percent

FROM (

  SELECT

    EXTRACT(YEAR

    FROM

      order\_purchase\_timestamp) AS year,

    EXTRACT(MONTH

    FROM

      order\_purchase\_timestamp) AS month,

    FORMAT\_DATE('%b %Y', DATE(ORDER\_PURCHASE\_TIMESTAMP)) AS time\_period,

    COUNT(order\_id) AS order\_count

  FROM

    `target.orders`

  WHERE

    order\_status = "delivered"

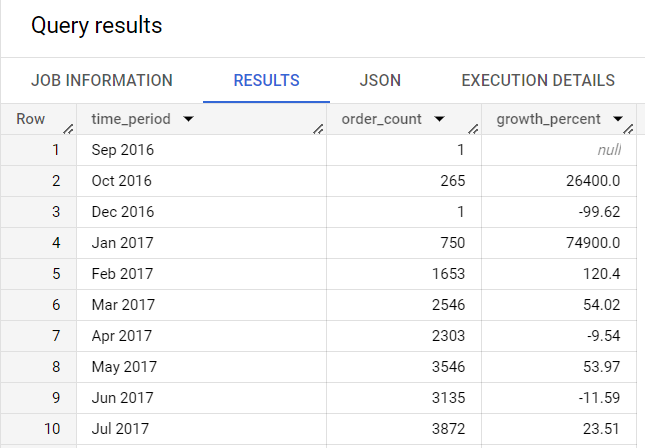
  GROUP BY

    month, year, time\_period) t1

ORDER BY

  year, month;

Result:



Can we see some seasonality with peaks at specific months?

No, as for some months data is showing orders of only 1 or 2 records and also there is huge spike in orders is seen in different months so we can/t comment on seasonality.

Overall business is in uptrend and sharp spike in orders in seen MoM basis.

2.2 What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query:

SELECT

  temp.purchase\_time,

  COUNT(\*) AS total\_orders

FROM (

  SELECT

    order\_id,

    CASE

      WHEN TIME(order\_purchase\_timestamp) BETWEEN "00:00:00" AND "07:00:00" THEN "Dawn"

      WHEN TIME(order\_purchase\_timestamp) BETWEEN "07:00:01" AND "12:00:00" THEN "Morning"

      WHEN TIME(order\_purchase\_timestamp) BETWEEN "12:00:01" AND "18:00:00" THEN "Afternoon"

      WHEN TIME(order\_purchase\_timestamp) BETWEEN "18:00:01" AND "23:59:59" THEN "Night"

    END

    AS purchase\_time

  FROM

    `target.orders`) TEMP

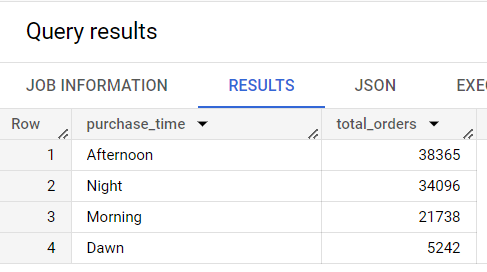
GROUP BY

  temp.purchase\_time

ORDER BY

  total\_orders DESC;

Result:



**Brazilian customers usually tend to buy in afternoon and night.**

**3. Evolution of E-commerce orders in the Brazil region:**

3.1 Get month on month orders by states

Query:

SELECT

  state , time\_period , total\_orders,

  LAG(total\_orders) OVER(PARTITION BY state ORDER BY year, month ) AS prev\_month\_orders\_count,

  ROUND(((total\_orders - LAG(total\_orders) OVER(PARTITION BY state ORDER BY year, month )) / LAG(total\_orders) OVER(PARTITION BY state ORDER BY year, month))\* 100,2) AS MoM\_percent\_growth

FROM (

  SELECT

    state,

    time\_period,

    year,

    month,

    COUNT(\*) AS total\_orders

  FROM (

    SELECT

      o.order\_id, o.order\_purchase\_timestamp,

      EXTRACT(YEAR FROM order\_purchase\_timestamp) AS year,

      EXTRACT(Month FROM order\_purchase\_timestamp) AS month,

      FORMAT\_DATE('%b %Y', DATE(ORDER\_PURCHASE\_TIMESTAMP)) AS time\_period,

      c.customer\_state AS state

    FROM

      `target.orders` o

    JOIN

      `target.customers` c

    USING

      (customer\_id)

    ORDER BY

       year, month) t1

  GROUP BY

    state, time\_period,year, month) t2;



3.2 Distribution of customers across the states in Brazil

Query:

SELECT

  customer\_state AS state,

  COUNT(\*) AS total\_customers

FROM

  `target.customers`

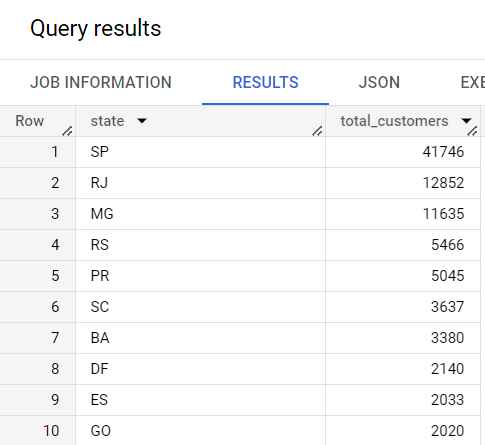
GROUP BY

  customer\_state

ORDER BY

  total\_customers DESC;

Result:



**4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.**

* 1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment\_value” column in payments table

Query:

SELECT

  \*,

  COALESCE((ROUND(((total\_orders\_value - LAG(total\_orders\_value) OVER(ORDER BY year))/LAG(total\_orders\_value) OVER(ORDER BY year))\* 100, 2)), 0) AS percent\_increase\_YOY

FROM (

  SELECT

    year,

    ROUND(SUM(payment\_value), 2) AS total\_orders\_value

  FROM (

    SELECT

      o.order\_id,

      o.order\_purchase\_timestamp,

      EXTRACT(MONTH FROM o.order\_purchase\_timestamp) AS month,

      EXTRACT(YEAR FROM o.order\_purchase\_timestamp) AS year,

      p.payment\_value

    FROM

      `target.orders` o

    JOIN

      `target.payments` p

    USING

      (order\_id)

    WHERE

      o.order\_status = "delivered")t1

  WHERE

    month BETWEEN 1

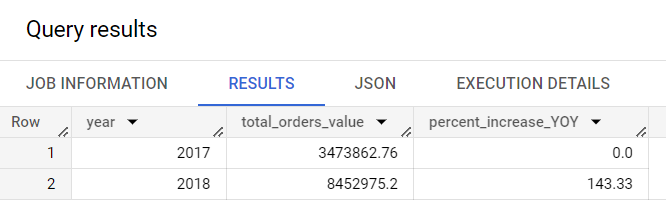
    AND 8

  GROUP BY

    year) t1

ORDER BY

  Year;



Comparison on Monthly total order value:

Query:

SELECT

  t2.Month\_n\_Year,

  t2.total\_orders\_value,

  ROUND(((t2.total\_orders\_value - LAG(total\_orders\_value) OVER(PARTITION BY month ORDER BY year))/LAG(total\_orders\_value) OVER(PARTITION BY month ORDER BY year))\*100, 2) AS percent\_increase

FROM (

  SELECT

    month,

    year,

    Month\_n\_Year,

    ROUND(SUM(payment\_value)) AS total\_orders\_value

  FROM (

    SELECT

      o.order\_id,

      o.order\_purchase\_timestamp,

      EXTRACT(month FROM o.order\_purchase\_timestamp ) AS month,

      EXTRACT(year FROM o.order\_purchase\_timestamp ) AS year,

      FORMAT\_DATE('%b %Y', DATE(ORDER\_PURCHASE\_TIMESTAMP)) AS Month\_n\_Year,

      p.payment\_value

    FROM

      `target.orders` o

    JOIN

      `target.payments` p

    USING

      (order\_id)) t1

  WHERE

    month BETWEEN 1

    AND 8

  GROUP BY

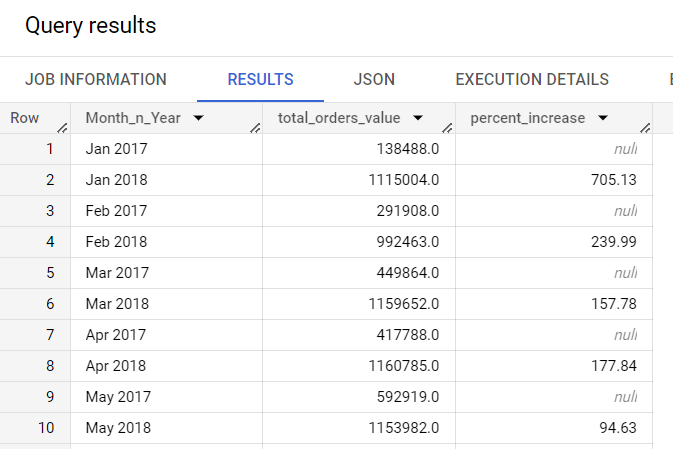
    month, year, Month\_n\_Year

  ORDER BY

    month, year) t2

ORDER BY

  month;



* 1. Mean & Sum of price and freight value by customer state

Query:

SELECT

  c.customer\_state,

  ROUND(SUM(oi.price)) total\_price,

  ROUND(AVG(oi.price)) avg\_price,

  ROUND(SUM(oi.freight\_value)) total\_freight,

  ROUND(AVG(oi.freight\_value)) avg\_freight

FROM

  `target.order\_items` oi

JOIN

  `target.orders` o

USING

  (order\_id)

JOIN

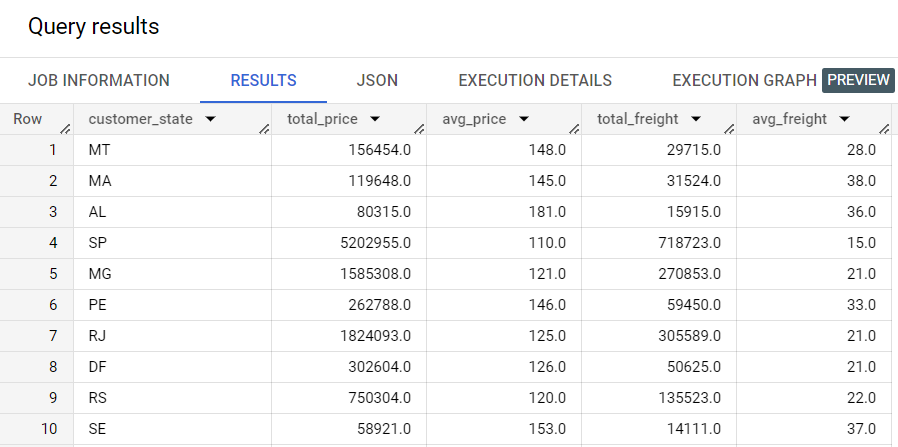
  `target.customers` c

ON

  c.customer\_id = o.customer\_id

GROUP BY

  c.customer\_state;



**5. Analysis on sales, freight and delivery time**

* 1. Calculate days between purchasing, delivering and estimated delivery

Query:

SELECT

  order\_id,

  TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS actual\_delivery\_time\_in\_days ,

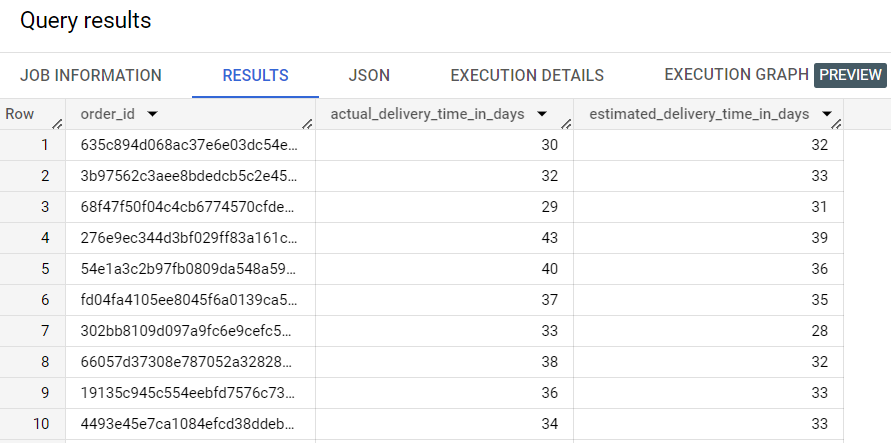
  TIMESTAMP\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, day) AS estimated\_delivery\_time\_in\_days

FROM

  `target.orders`

WHERE

  order\_status = "delivered";



* 1. Find time\_to\_delivery & diff\_estimated\_delivery. Formula for the same given below:

time\_to\_delivery = order\_delivered\_customer\_date-order\_purchase\_timestamp

diff\_estimated\_delivery = order\_estimated\_delivery\_date-order\_delivered\_customer\_date

SELECT

  TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS time\_to\_delivery,

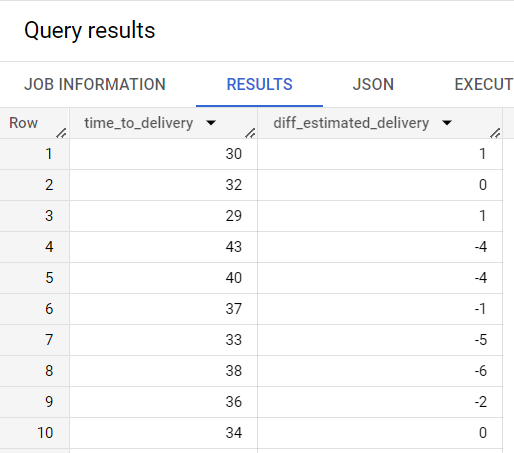
  TIMESTAMP\_DIFF(order\_estimated\_delivery\_date, order\_delivered\_customer\_date, day) AS diff\_estimated\_delivery

FROM

  `target.orders`

WHERE

  order\_status = "delivered";



* 1. Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

SELECT

  c.customer\_state,

  ROUND(AVG(oi.freight\_value),2) AS avg\_freight\_value,

  ROUND(AVG(o.time\_to\_delivery),2) AS avg\_time\_to\_delivery,

  ROUND(AVG(o.diff\_estimated\_delivery),2) AS avg\_diff\_estimated\_delivery

FROM

  `target.order\_items` oi JOIN

( SELECT

    order\_id, customer\_id,

    TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS time\_to\_delivery,

    TIMESTAMP\_DIFF(order\_estimated\_delivery\_date, order\_delivered\_customer\_date, day) AS diff\_estimated\_delivery

  FROM

    `target.orders`

  WHERE

    order\_status = "delivered") o

USING

  (order\_id)

JOIN

  `target.customers` c

USING

  (customer\_id)

GROUP BY

  customer\_state;



* 1. Sort the data to get the following: Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

States with Highest Avg freight value

SELECT

  customer\_state

FROM (

  SELECT

    customer\_state,

    ROUND(AVG(freight\_value),2) AS avg\_freight

  FROM

    `target.order\_items` oi

  JOIN

    `target.orders` o

  USING

    (order\_id)

  JOIN

    `target.customers` c

  USING

    (customer\_id)

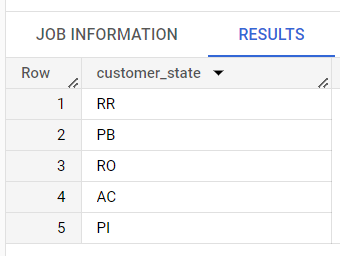
  GROUP BY

    customer\_state

  ORDER BY

    avg\_freight DESC) t1

LIMIT 5;



States with lowest Avg freight value

SELECT

  customer\_state

FROM (

  SELECT

    customer\_state,

    ROUND(AVG(freight\_value),2) AS avg\_freight

  FROM

    `target.order\_items` oi

  JOIN

    `target.orders` o

  USING

    (order\_id)

  JOIN

    `target.customers` c

  USING

    (customer\_id)

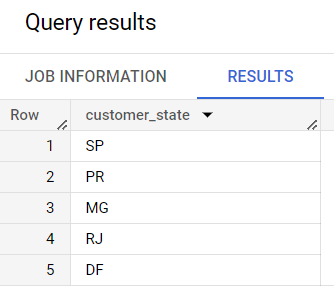
  GROUP BY

    customer\_state

  ORDER BY

    avg\_freight ) t1

LIMIT 5;



* 1. Top 5 states with highest/lowest average time to delivery

States with highest average time to delivery

SELECT

  customer\_state

FROM (

  SELECT

    customer\_state,

    ROUND(AVG(time\_to\_delivery),2) AS avg\_delivery\_time

  FROM (

    SELECT

      customer\_state,

      TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS time\_to\_delivery,

    FROM

      `target.orders` o

    JOIN

      `target.customers` c

    USING

      (customer\_id)

    WHERE

      order\_status = "delivered") t1

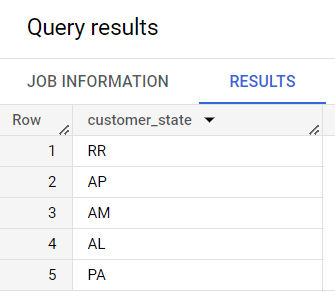
  GROUP BY

    customer\_state

  ORDER BY

    avg\_delivery\_time DESC) t2

LIMIT 5;



States with lowest average time to delivery

SELECT

  customer\_state

FROM (

  SELECT

    customer\_state,

    ROUND(AVG(time\_to\_delivery),2) AS avg\_delivery\_time

  FROM (

    SELECT

      customer\_state,

      TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS time\_to\_delivery,

    FROM

      `target.orders` o

    JOIN

      `target.customers` c

    USING

      (customer\_id)

    WHERE

      order\_status = "delivered") t1

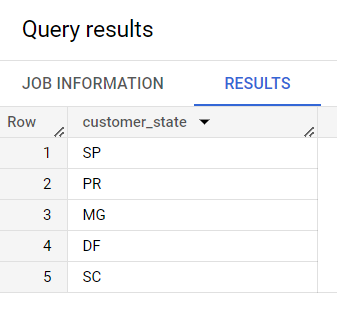
  GROUP BY

    customer\_state

  ORDER BY

    avg\_delivery\_time ) t2

LIMIT 5;



* 1. Top 5 states with really fast delivery compared to estimated date

SELECT

  customer\_state,

  ROUND(AVG(estimated\_delivery\_time - actual\_delivery\_time), 2) AS delivery\_time\_difference

FROM (

  SELECT

    o.order\_id,

    c.customer\_state,

    TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS actual\_delivery\_time,

    TIMESTAMP\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, day) AS estimated\_delivery\_time

  FROM

    `target.orders` o

  JOIN

    `target.customers` c

  USING

    (customer\_id)

  WHERE

    order\_status = "delivered") t1

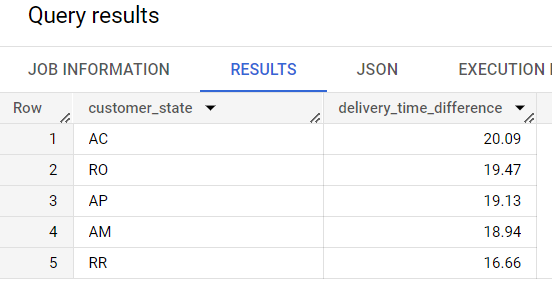
GROUP BY

  customer\_state

ORDER BY

  delivery\_time\_difference DESC

LIMIT 5;



states with not so fast delivery compared to estimated date

SELECT

  customer\_state,

  ROUND(AVG(estimated\_delivery\_time - actual\_delivery\_time), 2) AS delivery\_time\_difference

FROM (

  SELECT

    o.order\_id,

    c.customer\_state,

    TIMESTAMP\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) AS actual\_delivery\_time,

    TIMESTAMP\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, day) AS estimated\_delivery\_time

  FROM

    `target.orders` o

  JOIN

    `target.customers` c

  USING

    (customer\_id)

  WHERE

    order\_status = "delivered") t1

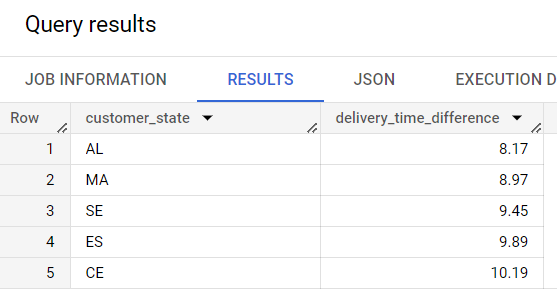
GROUP BY

  customer\_state

ORDER BY

  delivery\_time\_difference

LIMIT 5;



**6. Payment type analysis:**

* 1. Month over Month count of orders for different payment types

SELECT

  time\_period,

  payment\_type,

  COUNT(\*) AS total\_orders

FROM (

  SELECT

    p.order\_id,

    p.payment\_type,

    EXTRACT(YEAR FROM order\_purchase\_timestamp) AS year,

    EXTRACT(Month FROM order\_purchase\_timestamp) AS month,

    FORMAT\_DATE('%b %Y', DATE(ORDER\_PURCHASE\_TIMESTAMP)) AS time\_period

  FROM

    `target.payments` p

  JOIN

    `target.orders` o

  USING

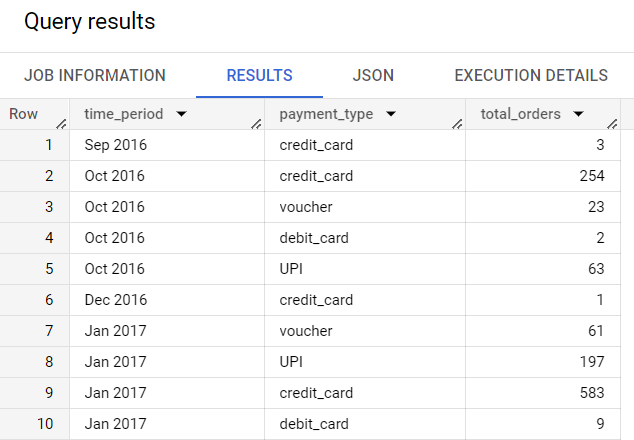
    (order\_id)) t1

GROUP BY

  time\_period, payment\_type, t1.YEAR, t1.month

ORDER BY

  t1.YEAR, t1.month ;



* 1. Count of orders based on the no. of payment installments

SELECT

  payment\_installments,

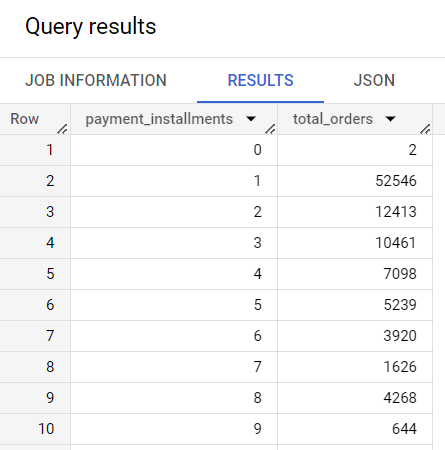
  COUNT(\*) AS total\_orders

FROM

  `target.payments`

GROUP BY

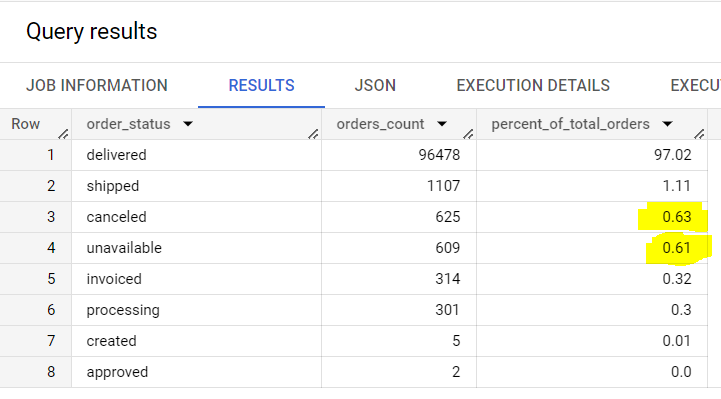
  payment\_installments;



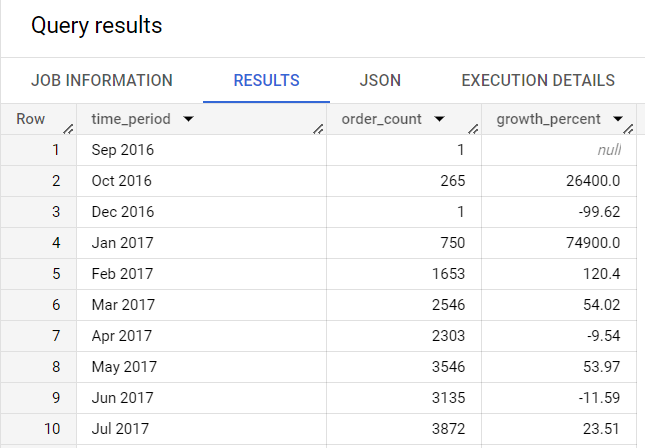
**7. Actionable Insights**

* Total 609 orders were unavailable and 625 orders were cancelled during the given time period, which makes it to be around 1.2 % of total orders.

We can reduce this number by studying the reasons behind order cancellation and items unavailability.



* We can see how the orders trajectory is showing very abrupt increase in orders volume with in very short time. Looking at overall trend, it is seen that business is picking up very fast in brazil so company has to be ready with extra workforce. To avoid high risk, it can consider hiring contractual employees.



* Company received low rating for maximum orders in highlighted states; need to study further about the reasons for customer dissatisfaction to such great extent in these states.

This is the query for counting the number of rating in each state.

SELECT

  \*

FROM (

  SELECT

    c.customer\_state,

    orv.review\_score

  FROM

    `target.order\_reviews` orv

  JOIN

    `target.orders` o

  USING

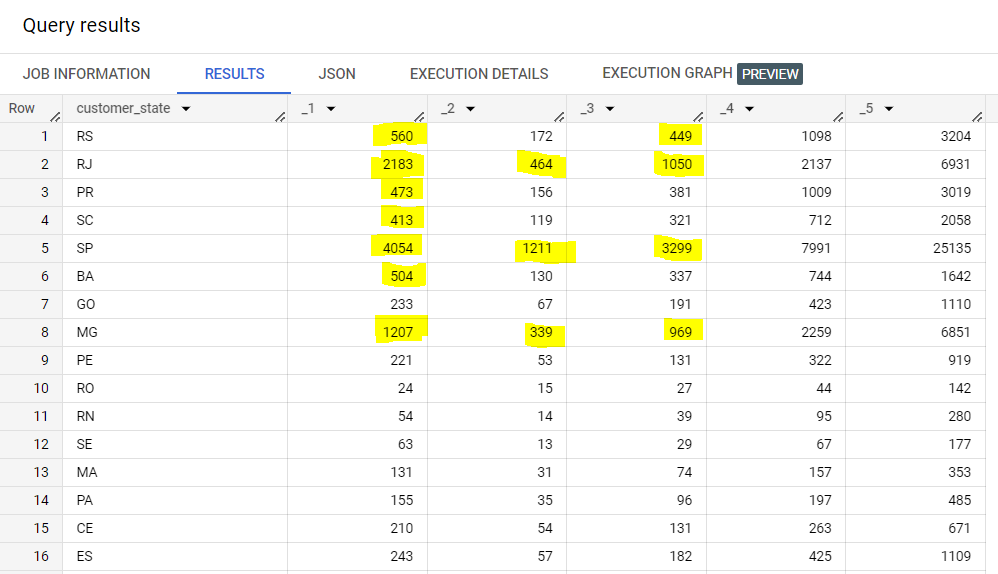
    (order\_id)

  JOIN

    `target.customers` c

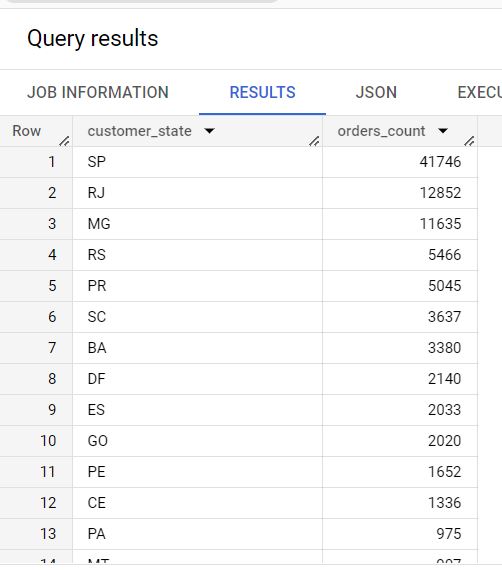
  USING

    (customer\_id)) PIVOT(COUNT(\*) FOR review\_score IN (1, 2, 3, 4, 5));



**8. Recommendations**

* As Brazilian customers usually tend to buy in afternoon and night, we can increase staff in during this time frame in order to manage the customers’ requests, and services better during this time by reducing workforce of morning and dawn.
* We can see, only 3 state contribute for maximum volume, and rest of the state need to be focused for improving the business.



* Avg delivery time is quite high for most of those states from where company is receiving quite less volume of orders, detailed study is needed further for checking the other reasons behind such low volume of orders from majority of states. Huge delivery time can be the one of the reason and need to work on it.

States with highest average delivery time -

